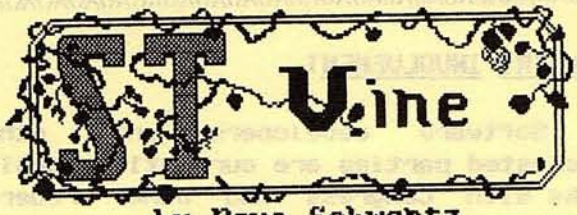


by Dave Schwartz
& Kim Beattie

The latest information from ATARI Corp. on release dates for their new hardware is as follows:

Laser printers - End of May (may be out by the time you read this)
Mega STs - End of June
ATARI PCs - June or July
1200 Baud Modem - May be out by the time you read this.

But this can change!

New announcements and releases:

From UNISON WORLD (PrintMaster Plus): NEWSROOM. This is a desktop publishing system currently available for the IBM PCs.

From WORDPERFECT CORP.: WORDPERFECT, the number one selling word processing package for the IBM PC, is expected to be shipped in June. This powerful package will include: GEM support, macros, spelling checker, thesaurus, math functions, merge, paragraph/outline numbering, sort, footnotes/endnotes, table of contents and index generation, and file compatible with WordPerfect 4.1 for the IBM PC and other computers allowing direct document transfer to the ST without losing document format.

From HYBRID ARTS: The long awaited MIDIMAZE (Kill a Happy Face) should be shipped anyday now as the ads for this product are now appearing.

From FTL: The also long awaited DUNGEONMASTER is now being sent to Beta Test and is planned for July release.

From TRUE BASIC, INC.: TRUE BASIC has been available for the IBM PCs and is available for the ST. Written by John Kemeny and Tom Kurtz, inventors of the original BASIC over 20 years ago, bring a new updated structured BASIC to the ST.

From QMI: DESKCART! is a plug-in cartridge containing 14 GEM desk accessories. These can be used in any GEM applications. The cartridge includes a battery operated clock and has the following accessories: calendar, notebook, card file, calculator, typewriter, address book, VT-52 terminal, control panel, screen dump, memory test, RAMdisk, disk utilities, and print spooler. Because DESKCART only takes up one slot in the desktop, you can also load 5 other accessories for a total of 19 desk accessories available to use.

From EI/O PRODUCTS: ARTABLET is a graphics tablet that comes in 3 sizes - 6x9, 12x12, 12x18. Each comes with an attached drawing stylus, power supply, RS232 cable, device driver for STs with TOS in ROM, and instructions and is compatible with DEGAS, DEGAS ELITE, NEO-CHROME, PUBLISHING PARTNER, CAD 3D, FIRST CAD, and AEGIS ANIMATOR.

That's all the news for now, see you next month.

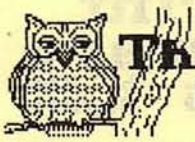
(SUA, cont.)

will get involved in our efforts. Your membership dues will be a great start. With your first edition of CounterPoint, you will be provided with all the information you will need concerning how to get further involved in this worthwhile project.

Membership dues are $15.00 per year and there are no other dues or fees. Your membership includes a free 1-year subscription to CounterPoint, the S.U.A. Quarterly and entitles you to participate in all of our programs and functions.

Presently, the S.U.A. supports only the Atari 8-bit and ST series computers. For further information or to obtain an original membership application and Users Summary, call (505)266-6234 or write today. To become a member, send check or money order to:

The Software Users Association
25076 Perimeter Drive
Albuquerque, New Mexico 87116



Thinking About Computers

by Bob Haynes

MAMA DON'T 'LOW NO LOOK-AND-FEEL HERE

At a recent APESIG meeting, your talented Newsletter Editor was treating the assembled multitude to an onscreen look at some code he had written in C. Some of the members were about to step outdoors for some real excitement - watching the grass grow - when he cut away from the listing and ran the program. Instant electrification. On the screen there appeared a well-known public domain graphics demo written in BASIC about five or six years ago. Talk about clones. Mr. Stomp's creation looks like the original. It moves like the original. It sounds like the original. It loads to the screen like the original. The Doublemint twins are easier to tell apart. But it is written in a language as different from BASIC as Italian is from English.

At the same meeting, ACCESS Treasurer Kim Beatty made favorable mention of an article in COMPUTE! magazine. The article showcased a half dozen versions of a program, each written in a different ST language.

As anecdotes, the foregoing instances won't draw any chuckles. But they are relevant to the latest wrinkle in the Piracy Wars: Look-and-feel. The software industry needs to throw a necktie party for the idiot child who came up with this non-concept. In case you have been vacationing in Bora Bora the past six months, our peerless courts have ruled that a program that looks and feels like an already-published program violates copyright laws. Period. Beginning of (prison) sentence. So, we have Lotus suing a couple of companies on the grounds of look-and-feel. In turn, Lotus is being sued for looking and feeling like VisiCalc.

Guys! Guys! Listen up! The time to turn upon your lawyers is now. Throw the rascals out. Don't let them do this to you - sue one another into bankruptcy, that is. The supreme irony may come when a clone maker sues an "original" programmer for copying the clone's improvements.

Let's face facts. Look-and-feel is one monumental hunk of legal Swiss cheese. Start with the front end. Now that they have been alerted, clone makers will find that changing the look of a program is surpassing easy. A different color background, an altered character set, two-column menus (rather than one), a rearranged order of items on the menu, changes in nomenclature. Thus, "Press SPACEBAR to continue" becomes "HIT escape TO GO ON." And so on, almost ad nauseum. As to the "feel" of a program, what does that nebulous term mean, exactly? Input method? Order of commands? Output or product? Bells and whistles? Documentation? As Humpty Dumpty said to Alice in Wonderland, "Words mean whatever I say they mean." Humpty Dumpty was obviously a judge.

No matter. A couple of days' work by a hack programmer can change the "feel" of a program to the point where the "original" programmer can't recognize her/his own brainchild.

Recall the clone mentioned at the beginning. We have a plainly absurd state of affairs. Your editor would be subject to a lawsuit if his "copy" written in entirely different code were not in the public domain. But the clone maker who inserts a few cosmetic changes without altering the essence of the code would go scot free.

Another joker in the deck is the word and. It's surprising that some defense attorney somewhere has not grabbed hold of the significance of that little connective. A plaintiff must prove that both look and feel have been ravished by the brutish clone maker. This renders the clone maker's job twice as easy. Change the "look"; change the "feel". Take your choice. And the plaintiff's case joins the baby alligators in the sewer. If I seem to

(Continued on next page.)

(Thinking, cont.)

have a soft spot for clone makers, that's because I do. Why? Because clone makers have improved the breed in both software and hardware. They have helped to force prices down to what they should be. And they have provided competition that has forced the originators to improve their own wares much faster than they wouldotherwise have done so. (As a footnote, isn't competition what made America great? Isn't it UNAMERICAN to stifle competition - as the software and hardware pros and their legal hitmen are trying to do?)

We know that clones are superior to the "original" products, either in lower price or in quality - usually both. We know this because the makers of the "original" products claim that the copies are inferior. Note the consistent use in this piece of quotation marks around the word "original". About 99 and 44/100% of all "original" program code is derived from earlier sources. But do these sanctimonious hypocrites acknowledge their debts to their sources? Is the Pope a Buddhist?

**Programming challenge: Create a program that looks like an existing program and feels like that program - but does something altogether different.

**Question to ponder: Springboard has openly stated that it will not release an Atari version of Newsroom. Could Springboard sue a company that published a Newsroom clone for Atari? Could they lose the case?

[One thing I would like to add: it is to the advantage of the user if different programs of the same type have the same command structure, similiar menus, etc. It makes life easier if he doesn't have to learn a completely new set of commands. Y'Know, two different models of car with standard transmissions have pretty much the same "feel" - steering wheel and gearshift are in the same place - thank God! Is there a case for a lawsuit there? - Ed.]

Print Library News

by Bob Haynes

As announced at last month's meeting, the finals of the Egregiously Ugly Color Contest (EUCC) will be held during the June 24 meeting. Attend, and see who will be crowned MVP (Most Vile Pigmenter). Better yet, bring your own repulsive entry. The rules:

-)1. Create a screen display using at least two colors on a String One (8-bit) or String Two (16-bit) Atari. Bring a copy of your masterpiece to the meeting.

-)2. Use any programming language. However, languages other than BASIC or M/L mube presented as runtime programs. Entries will be displayed on color monitors, not on TV screens.

-)3. Entries will be judged only on the loathesomeness of the color combination. Grotesque subject matter, lack of artistic merit, and woo-woo sound effects won't mean beans in the judging.

-)4. The decision of the judges will be final.

The winner will receive a computer peripheral, complete with documentation and packaged in suitably tasteless fashion. All entries will be eligible for inclusion on future club disks. Your chance awaits for fame and possible fortune.

* * *

On other news fronts, the renewed interest in programs from the defunct Softside magazine will be rewarded at the June meeting in the form of documentation for many of the programs. Some of the early members of ACCESS (back in the caveman days) learned to progam with Softside.

Because of the riotous enthusiasm of some club members, an updated selection of game tips will also be available. These have been offered in the past, but fell by the wayside due to indifference. Times, it would seem, change.

The Polyglot Programmer

by Michael Stamp

BIT TWIDDLING

This refers to manipulating the individual bits in a byte - or several bytes, depending upon the language used. The operators used to achieve this are variously called masking, Boolean, or bitwise operators. The are found in most computer languages, with the notable exception of Atari Basic. The only ways I know of to do the same thing in that language is by adding some USR routines, or by some rather clumsy arithmetic operations. Sorry.

There are three masking operators: "and", "or", and "exclusive or" (xor). Don't confuse these with the logical operators "AND" and "OR", which operate between two relational expressions, such as:

```
X<1 OR X>10
```

The masking operators are used on two bytes (or words or numbers, depending upon the language) and operate on the corresponding bits of each, independent of the other bits. Thus, the number of bits in the "objects" is relatively unimportant, and, to keep the examples short, I will use 4-bit bytes. (Why play favorites?) Now for some terminology: usually one byte is a variable, called the DATA, and the other, called the MASK is controllable. You choose the MASK to control which bits are to be affected. Unfortunately, there is not agreement among various languages on what symbols to use for the operators. In the three languages that I am familiar with the following symbols are used:

OPERATOR	BASIC XL	C	ASSEMBLY
and	&	&	AND
or	!	|	ORA
eor	%	^	EOR

The operators can best be defined by giving their multiplication tables.

and	DATA 0	DATA 1
MASK 0	0	0
MASK 1	0	1

The result bit is 1 only if both the DATA bit and MASK bit are 1. Notice that if the MASK is 1 the result is just a copy of the DATA, while if the MASK is 0 the result is 0, no matter what the data is. Thus, "and" can be used to force a bit to 0. For example, if MASK = 1011, then MASK $ DATA will have bit #2 set to 0, with all the rest of the bits left as they were. 1011, then, is the bit #2 "and" mask. If you reverse all the bits in MASK you get 0100; this is called the "complement" or "ones complement" of MASK. Notice that in DATA & 0100, all the bits except bit #2 will be 0, while bit #2 will be whatever it is in DATA. This is the way to "read" a particular bit. Used as a logical relation, in an IF statement for instance,

```
IF DATA&4 THEN ...
```

the test will be true if bit #2 is 1 and false if it is zero. (I've used "4" for 0100, since most languages don't allow binary representation.)

or	DATA 0	DATA 1
MASK 0	0	1
MASK 1	1	1

The result of "oring" two bits together is 1 if either the DATA or the MASK is 1. Notice, that if the MASK is 1 the result is 1, whatever the DATA is, while MASK=0 just results in a copy of the DATA. Thus, "or" is used to force a bit to be 1. The "or" MASK to control bit #2 is 0100, which is just the complement of the "and" MASK.

(Continued on next page)

(POLYGLOT, cont.)

		DATA	
xor		0	1
MASK	0	0	1
	1	1	0

In the "xor", if the MASK is 0 the result is identical to "or". But, if the MASK is 1 the result is just the opposite of the DATA. Thus, "xor" can be used to "toggle" a bit - make it the opposite of whatever it was. Thus, with a MASK which is all 1's, "xor" with it will complement the DATA. (We will use this later.)

To summarize then, "and" is used to force a bit to 0 or to read a bit; "or" is used to force a bit to 1; and "xor" is used to "toggle" a bit. That's all there is to it! Since the bits are affected independently of each other you can combine MASKs by just adding them together to work on several bits at one time.

Now for an example: the following is a small routine that I found in the May, 1982 issue of the ACCESS Key. I've rewritten it in Basic XL to make use of the masking operators.

```
5 Dim Lu(3):Lu(0)=$40
6 Lu(1)=0:Lu(2)=$20:Lu(3)=$60
7 Charset=57344:Rem ROM CHARACTERS
10 Graphics 4+16:Open #1,4,0,"K:"
20 D1=Dpeek(560)
30 Mem=Dpeek(D1+4)
40 Trap 40:X=0:Chr=33:Goto 90
50 If X)410 Then ? #6;Chr$(125):X=0
60 Get #1,Chr
70 Index=Int((Chr&$60)/32)
75 Chr=(Chr&$9f)!Lu(Index)
80 Mask=0:If Chr&$80 Then Mask=$ff:
   Chr=Chr&$7f
90   For I=0 To 7
100    Poke Mem+X+10*I,
       Peek(Charset+Chr*8+I)%Mask
110  Next I
120 X=X+1:If Int(X/10)=X/10 Then
    X=X+70
130 Goto 50
```

This routine allows you to type in BIG characters on a Graphics 4 screen. It does that by Poking the character shape from the ROM character set right into screen memory. (Line 100) Now, to locate the character in the table you use as an index, not the ATASCII number, but the screen code. (For more on this subject, see "What a Character", ACCESS Key, 10/85.) The two are related by the rule:

> "If X)127, subtract 128; then, if X<32, add 64; or if 32<X<95, subtract 32; then add back 128 if you subtracted it before.!

The process can be simplified if you notice that bits #5 and #6 are the only things you need to change. That's what this routine does in lines 70-75. Those bits are extracted by &$60 ($60 = 96 = 32 + 64), moved to the right of the byte by dividing by 32. The result is used as an index into the look-up table, Lu, which contains the correct bits which are put back by the "or". Bit #7 is read with the "and", if it is one (an inverse character) MASK is set to all 1's and bit #7 is set to 0. In line 100 the chatacter shape, Peeked out of the ROM set, is "xor" with MASK. If the character is an inverse this will toggle them; otherwise, they are left alone.

This conversion routine in lines 70-75, together with the definition of the array, Lu, can be pulled out and used anytime you want a routine to convert ATASCII to screen code. It's fast and clean. And so useful that I thought I'd give it too you in two other languages.

First, 6502 Assembly Language:

```
          *= ????  ;NOT relocatable
CONVERT   PHA      ; save character
          ROR
          ROR      ; shift 5 places to
          ROR      ; right. Same as
          ROR      ; dividing by 32
          ROR
          AND #3   ; mask bits
          TAX      ; index into LU
          PLA      ; restore character
          AND #$9F ; strip off bits
          ORA LU,X ; put in new bits
          RTS      ; Done!
LU        .BYTE $40,$00,$20,$60
```

(Continued on next page.)

XM301 WARNING

From The POKEY PRESS! 1987
by Paul Alhart

(Reprinted from ZMAG, #49, 4/27/87)

If you own an XM301 modem, you may own an electronic "Time Bomb". After a rash of hardware failures last month, which included smoking a disk drive and two printer interfaces, I found the cause of my problem to be my XM301. The modem worked fine, but was killing off my system piece by piece.

The reason has to do with the thirteen wires coming from the serial I/O plug, although only nine wires are actually used by the modem. The other four wires have about 1/8 inch bare wire showing, and are just hanging around, unterminated, waiting to touch something they shouldn't. I have checked other XM301 modems and this condition existed in them too.

Here is what to do immediately:

With all power off, remove the two screws from the bottom of the modem and lift off the plastic case. Inspect the wires where they enter the modem. You will find that four of the wires are not connected to anything. If these wires have any bare metal showing, cut it off. Be careful to keep the cut-off pieces from falling into the modem. Next, tape each wire individually, so that it cannot possibly touch any other wires or parts in the modem. Put the modem back in its case, replace the screws, and you are done. I have written to Atari regarding this problem, but have not received a reply as yet.

[Note: This may be an isolated problem, but when I checked my modem I found bare wires looking for trouble. I found heat shrink tubing worked best. If you own an XM301, I highly recommend checking for this potential disaster now! - ZMAG Ed.]

(POLYGLOT, cont.)

To use this subroutine, just place the character to be converted in the Accumulator, JSR to CONVERT, and upon return the screen code will be in the Accumulator. Only the A and X registers are used. (I have a longer version of this which will convert a whole string at one time.)

Now in C:

```
10 convert(c)
20 char c;
30 $(   char index,*lu ;
40      lu="\64\0\32\96";
50      index= (c )) 5) &3 ;
60   return (c | lu[index]) ;
70 $)
```

(C doesn't use line numbers; I put them there just to refer to.) The index is found in line 50 just as in Assembly Language. "))" means shift right; the 5 tells how many bits to shift. "&3" picks off the bits we want. In line 60, the new bits from the table, lu, are "ored" with c and the result is returned. Use this routine in c as:

```
char b,c ;
b= convert(c) ;
```

Notice how similiar the routines are in the three languages - just small differences in syntax. That's usually how it is: the hard part is figuring out the algorithm (what you want to do and how to do it). The actual coding then becomes a detail.

Homework problem: Write a similiar routine that will convert from screen code to ATASCII. Code it in all three languages. [Hint: consider how the table LU would change.]

"Friendly no, but it *is* user tolerant."

ACCESS KEY EDITORIAL POLICY

The ACCESS KEY is published monthly by the Atari Computer Club Encompassing Suburban Sacramento. It is mailed free to all club members. Material submitted for publication should be in the hands of the Editor on or before the deadline, which is listed under "FUTURE CLUB MEETINGS". Submissions for inclusion in this newsletter are actively solicited. Additionally, if items of general or special interest to our membership are discovered in newspapers or magazines, you are encouraged to send copies to the editor or simply notify him of the article. Your contributions are always welcome.

If you have a modem, you may submit material - text files, graphics screens, etc. - by uploading them to the Club's BBS. Be sure to leave a Message to the SYSOP giving him the names of the files.

ADS: Ads of a noncommercial nature are free to all ACCESS members. Commercial advertising rates are:

Full page	- $35
Half page	- $20
Quarter page	- $12

ACCESS is an independent computer club and user's group, and has no connection with Access Software, Inc., Atari Corporation, Atari Games, Inc., or any other organization or corporation of homonymous namesake. Opinions expressed within are those of the individual authors and correspondents and do not necessarily represent nor reflect the opinions, hopes, aspirations, or dreams of the ACCESS membership, its board, or any other sentient being (but it may!). Atari is a registered tradewark of Atari Inc.

MAIL: All mail should be sent to the official address:

A.C.C.E.S.S.
P.O. Box 1354
Sacramento, CA 95806

ACCESS.BBS

(916) 423-1316

SysOp: Dale Mauch

ACCESS
P.O. BOX 1354
SACRAMENTO CA 95806

SLCC JOURNAL
C/O NEWSLETTER EXCHANGE
PO BOX 1506
SAN LEANDRO, CA 94577-0374